BIT-LET

ötlet SZÁ 1985 SZEPTEMBER 24

ÉS AKIK NEM PROGRAMOZHATNAK



Ismét egy új tanév. Nem írhatom, hogy kezdődik, hiszen már néhány hete folyik a tanítás. Vajon mit hoz ez az új év a számítástechnika oktatásában? Mit hozhat egyáltalán? Tudjuk, hogy az általános iskolai program beindul, tudjuk, hogy a középiskolai folytatódik. Tudjuk, hogy egyre többet értenek a dologhoz azok a pedagógusok is, akiknek a kezében vagyon a dolog. Tudjuk, hogy a középiskolai szakkörök, s újabban az általános iskolaiak is egyre fantasztikusabb eredményeket érnek el, tagjaik egyre fantasztikusabb teljesítményekre képesek. Mégis, a szerkesztő már megint böstörög. Már megint talál valamit, ami nem jól van. Néhány évvel ezelőtt, amikor a program igazából beindult, egy kerekasztal beszélgetést szerveztünk, ahol épp a középiskolai oktatás volt a téma. Ott a jelenlévők – talán törzsolvasóink még emlékeznek is rá – két pártra szakadtak. Az egyik fél azt mondta, hogy a középiskolai oktatás feladata, hogy megtanítsa a diákoknak a számítógép használatát, s esetleg a legjobban érdeklődőknek lehetőséget adjon a programozás alapjainak elsajátítására. A másik fél ezzel szemben azt követelte volna, hogy legyen a programozás néhány órában mindenki számára kötelező tantárgy, mert ez komoly szemléletformáló hatással bír, s egyébként sem egy ördöngösség. A vita nem dőlt el, nem is dőlhetett el. Az iskolák gyakorlata azonban az elvek szintjén az első álláspontot tükrözi. Ez végül is érthető, hiszen az az iskolánkénti 1–2 gép nem elegendő másra. Amiért ez a kis írás most íródik – épp az, hogy vajon azt a bizonyos első álláspontot tükrözi-e igazán az iskolákban folyó munka? Megvalósul-e a cél, hogy tudniillik minden diák legalább felhasználói szinten jusson a számítógéphez. Nos, épp ez az, hogy távolról sem. Adataink nincsenek róla, de amúgy saccolásos módszerrel különösebben nagy bátorság nélkül kijelenthetjük, hogy a középiskolákban végző diákok legalább ötven százaléka ma még úgy hagyja el a középiskolát, hogy soha nem látott közelről számítógépet, hogy amennyiben gép elé kellene ülnie éppolyan zavarba jönne, mint a nagymamája. A szerkesztő inkább azokkal ért egyet, akik azt mondják, hogy az iskolákban akkor lehetne igazi szemléletformáló erejű számítástechnikai oktatást végezni, ha lenne lehetőség a néhány órás kötelező programozói tárgy bevezetésére. (Hogy ez mennyire fontos, azt a szerkesztő már a gyakorlatban is tapasztalta. Felnőtt emberekkel is csak úgy sikerült megértetni, hogy hogyan tartja kezében a programozó a gépet, hogyan tudja azt mondani a gépnek, hogy ezt is csinálja meg, meg azt is, hogy egy rövid programot ízeire szedve néhány órában bemutattuk nekik, hogy is kell programozni. Ettől ők még nem lettek programozók, nem is kell, hogy azok legyenek, de legközelebb már nem mondják azt a bemutatást vezetőnek, amit az ezt az oktatást megelőző bemutatón, hogy „– Mondja meg a gépnek, hogy még rendezzen névsorba is!".) A szerkesztő azonban tudomásul veszi a realitásokat, s belenyugszik, hogy nincs programozás. De abba már nem tud belenyugodni, hogy a középiskolákban mindössze néhány tucat tehetséges diák jusson hozzá a gépekhez. Rendben van, az ő programozási tudásuk érték, nekik több lehetőséget kell biztosítani, hogy géphez üljenek. De ne felejtkezzünk meg a többiekről sem! Adjunk lehetőséget, alkalmat nekik is!

Igen ám, mondhatom én, meg mondhatja még rajtam kívül több tucat újságíró, pedagógus, hogy ez fontos. De ne feledjük, hogy az iskolák számítástechnikai munkáját mindig a legjobbak teljesítménye alapján ítélik meg. Annak alapján, hogy milyen programokat küld be az iskola szakköre a különböző pályázatokra, meg annak alapján, hogy a különböző bemutatókon hogyan szerepelnek a diákok. Így hát egy olyan iskola, amelyikben van két kiválóan programozó diák, s a számítógépet rajtuk kívül még senki sem látta, jobb színben tűnhet föl az oktatásügy irányítói előtt, mint az, amelyikben nyolcszáz diákból 799 találkozott már a számítógéppel, tudja kezelni, érti, hogy mire való a gép, mit lehet vele csinálni, de egyikük sem mélyült bele a dologba annyira, hogy különösebben érdekes programot tudna írni. Márpedig az oktatásügy irányítóin sok múlik. Befolyásolja a tanár prémiumát, fizetését, hogy mi a róla kialakult vélemény, s nem feledhetjük azt sem, hogy az új gépek elosztásánál is a kialakult értékítélet a döntő. Jó lenne tehát, ha az új tanévben több figyelmet kapnának a programozni nem tudók, ha az ő képzésük, az ő szemléletkialakításuk nagyobb súllyal esne latba a tanári, iskolai munka megítélésénél.

Angyalosi László

BELÜLRŐL

ötlet

HÍROLDAL

Fordítógép

Minden eddiginél megbízhatóbb, hűebb fordítást végző és gyorsabb japán–angol nyelvű fordítógépet készített a japán Toshiba cég. Az óránként ötezer szót lefordítani képes, mikroszámítógépes berendezés kiválóan alkalmazható a tudományos szövegek és műszaki dokumentációk esetében is. A szuperfordító még ez évben piacra kerül. Fejlesztői mintegy negyvenezer dollárt kérnek majd darabjáért.

Lézeres gép

Egy kis elektronikus kalkulátor méretével azonos nagyságú mikroszámítógépet fejlesztettek ki egy tallinni kutatóintézetben észt szakemberek. A rendkívüli kis méretet azzal érték el, hogy egy teljesen új elven működő, lézeres konstrukciót dolgoztak ki. Lézersugárral írják be és tárolják az információkat egy speciális tárolóelem molekuláiban. Szovjet kutatók már mintegy tíz éve kidolgozták a lézeres információbevitel, tárolás, átvitel módjára és a felhasználható anyagokra vonatkozó elveket.

IC-k

Aligha gyártottak valaha is olyan terméket a világon, melynek ára annyira zuhant volna, mint az integrált áramköröké. Mégis a gyártásukkal foglalkozó nagy cégek egyre nagyobb profitot tehetnek zsebre évente. A három elektronikai nagyhatalom az Egyesült Államok, Japán és Nyugat-Európa piaci forgalma nagyrészben integrált áramkörökből 1984-ben elérte a 23,5 milliárd dollárt, ami 1983-hoz képest 25 százalékos emelkedést jelentett. Az ez évi forgalom növekedése hasonló mértékben várható.

Sinclair gond

Sir Clive Sinclair a róla elnevezett nagysikerű, olcsó mikroszámítógépek kifejlesztője – bár például a múlt évben közel nyolcszázezer személyi számítógépet adott el – átmeneti anyagi gondokkal küzd. Hátrányosan érintette, hogy ez évben visszaesett a gépek iránti kereslet. Ezen még az sem segített, hogy Spectrum típusú gépe árát 180-ról 130 fontra csökkentette. Így további ambiciózus fejlesztő munkáihoz mintegy 15 millió font kölcsönt kíván felvenni.

Brazíliában került...

A címbeli mondat szerepel a Brazíliában forgalomba hozott mini- és mikroszámítógépek nagyrészén. A közelmúltban újabb nyolc évre meghosszabbították Brazíliában a híres számítógépi piacvédő törvényt. A törvény továbbra is nyílt utat ad a hazai számítógép-fejlesztőknek. Az első brazil mikroszámítógépek 1980-ban jelentek meg a piacon. Ma mintegy 16 000 embert foglalkoztatnak a latin-amerikai országban a számítógépiparban.

Agyműtét

A számítógéppel vezérelt robotkarok számtalan az ember számára nehéz, veszélyes vagy igen fárasztó munkát elvégeztek már eddig is. Újdonságként vehető a legfrissebb információ, amely szerint Kaliforniában egy beteg agyából egy mikroszámítógéppel és egy rétegvizsgáló berendezéssel összekapcsolt robotkar vett szövetmintát. A művelet jelentősége abban van, hogy a robot az embernél pontosabban és biztosabban képes elvégezni a műtétet.

Rendőr chip

Közismert, hogy milyen hatalmas károkat okoznak szerte a világon a számítógépes bűnözők. Ezek a károk csak növekedtek azzal, hogy a személyi számítógépek nagy száma kapcsolódik a központi számítógépekből kiépített nagy hálózatokra. Az illetéktelen információhoz jutást különféle hardver- és szoftvermegoldásokkal igyekeznek kiküszöbölni. A legújabb védekezési módot az amerikai INTEL cég „rendőr" chipje, az úgynevezett KEPROM (Key – azaz kulcs – és az EPROM szó összetételéből) biztosítja. A speciális chip egy 64 bites kulcsszó segítségével megoldja, hogy csak az illetékességgel rendelkező terminálok csatlakozhassanak a központi számítógépre. Az adatátviteli vonalak mindkét végén van egy-egy ilyen újraprogramozható KEPROM, amelyek az olvasási művelet előtt megbizonyosodnak arról, hogy kulcsszavaik azonosak. Találgatással a kulcsszóra ráhibázni szinte a lehetetlennel egyenlő és így kizárt a számítógépes bűncselekmények elkövetésének lehetősége.

Avis térkép

Új szolgáltatást vezetett be idén az Avis gépkocsikölcsönző világcég. Angliai, írországi, spanyol és nyugatnémetországi telephelyein számítógépes térképet adnak át más országokból érkezett ügyfeleiknek. A térkép eligazítja a gépkocsivezetőt, elősegíti a legtakarékosabb üzemanyagfogyasztást. Az ügyféltől származó, az egyéni érdeklődésre vonatkozó adatok alapján a számítógép speciális méretre „szabott" pl. színházat, koncerteket, vidám parkokat, stb. érintő útvonaltérképet is készít.

Étterem

Az egyesült államokbeli Palo Altóban egy IBM számítógépre alapozott automata éttérem működik. Minden asztalba, melyekhez a vendégek leülnek, be van építve egy billentyűzet. A megrendelt ételt, italt a pincér beüti az asztal billentyűzetén. Az adatok azonnal megjelennek a konyhában és elkezdődhet a rendelés elkészítése. A rendszer visszajelzi az étel elkészültét is, sőt az asztalba szerelt kijelzőn megjelenik az összfogyasztás ára is.

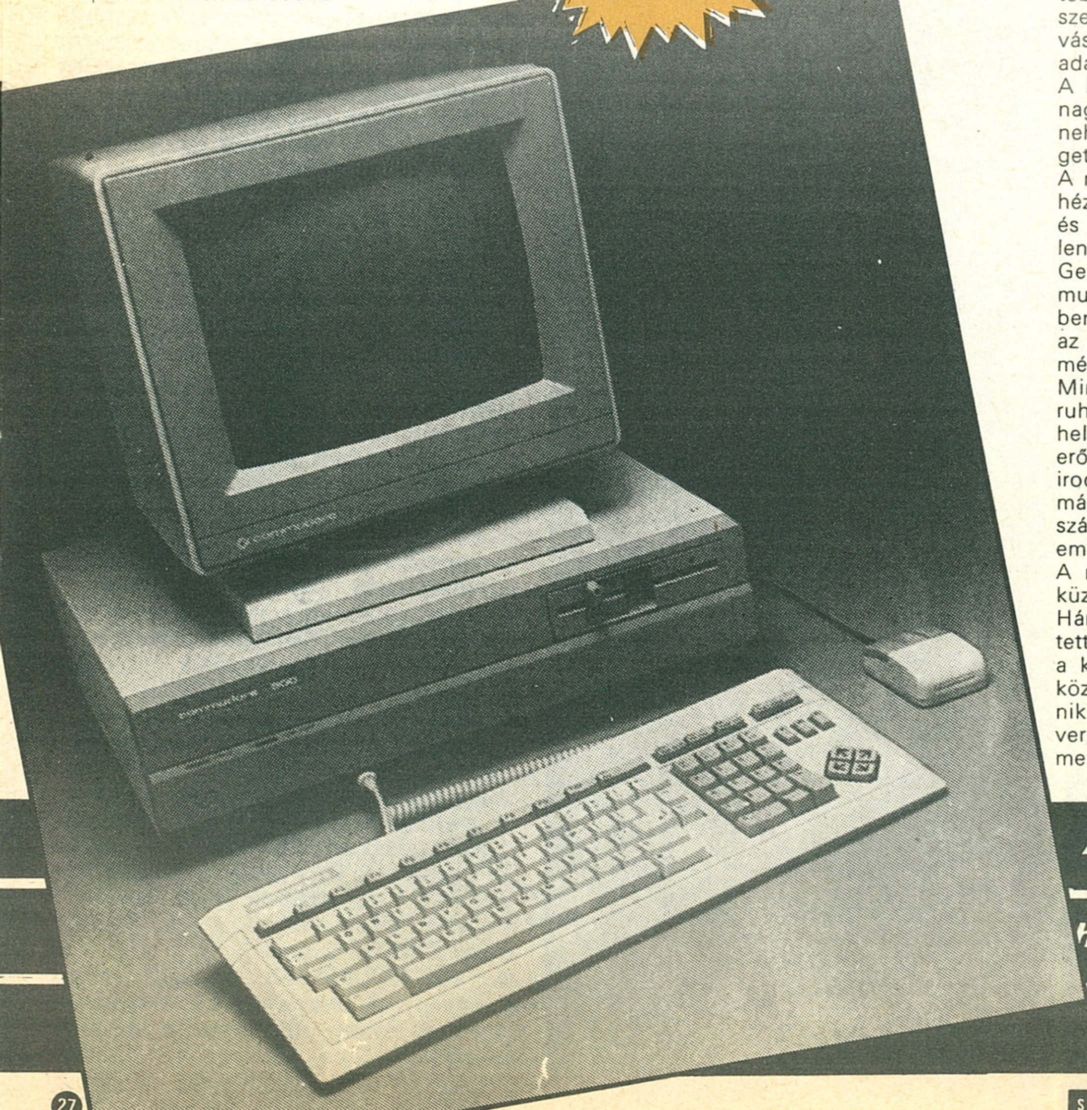
Chip teszt!

Az új összetett mikroprocesszorok tesztelése komoly problémákat okoz az Intelnek és a Zilognak egyaránt. A Zilog például már 1983 nyarán bejelentette a Z80000 típusú 32 bites processzorát. A szállítások kezdetét 1984. második felére tűzték ki, de a hibakeresés, tesztelés nehézségei miatt ez nem várható 1986 előtt.

Az Intel legújabb processzorában, a 80286-osban hiba jelentkezett és ez késleltete a Digital Research munkáját a 286-os konkurrens operációs rendszeren. A hibát, a 80286-os harmadik verziójában fedezték csak fel. Az Intel leállította a 432-es chip gyártását, de erőteljesen fejleszti az új, 80386-os chipet.

A Commodore International a hannoveri vásáron mutatta be új, többfelhasználós, UNIX-szerű rendszerét és egyedi munkaállomását. Az új számítógép – meglepően olcsón – 4000 dollár alatt lesz kapható.
A munkahely képernyője 14 inches, 1024x 800-as felbontású. A videóvezérlő 128 Kbyte RAM-ot használ és Bit-BLT technológiát fog alkalmazni.
A processzor egy Z8000, minimum 512 Kbyte RAM-al. A többfelhasználós, többtaszkos rendszer az ATZT UNIX VII rendszerével kompatibilis. A rendszer alapkiépítésben tartalmaz egy 20 Mbyte-os Winchester és egy 1–2 Mbyte-os floppy háttértárat. Opcionálisan kapható hozzá streamer (menő-szalag), második floppy meghajtó, vagy 40 és 67 Mbyte-os Winchester tároló. A felhasználó által használható memória 2 Mbyte lehet maximum. Európában a forgalmazás ez év szeptemberében kezdődik.





Commodore tábor

Sokféle számítástechnikai tábort rendeztek már hazánkban, de olyat, ahol minden táborlakónak jutott volna egy számítógép, aligha. Most ilyenre kerül sor Vépen, az Endrődy kastélyban. A táborban nyolc turnusban, mintegy hétszázhúsz gyermek ismerkedhet meg a számítástechnikával. A gépeket az angol Commodore és hazai partnere a Novotrade RT bocsájtja rendelkezésre. Az angliai táborok mintájára megrendezett tanfolyamokon Commodore 16-os gépeken gyakorolhatnak a gyerekek.

Piaci krízis..

Az Egyesült Államokban, az utóbbi hat hónapban erős visszaesést tapasztaltak a számítógép-forgalmazók. Míg 1981–82-ben a számítástechnika piaca „recesszió-álló" volt, úgy tűnik, ebben az évben ez a piaci szegmens is beállt az általános beruházási színvonalra. Az Egyesült Államokban a nem katonai jellegű termelőeszközök vásárlása márciusban 8%-kal esett, áprilisban további 7%-kal. A számítógép-beruházások ennél is gyorsabban estek vissza. Az IBM például áprilisban 30%-kal kevesebb rendelést kapott, mint az előző hónapban.
A Data Resources nevű cég prognózisa szerint a számítógépes piac növekedési üteme az 1984-es 16%-ról 3%-ra esik vissza az idén.
Az amerikai számítógépipar minden reménysége a személyi számítógép eladások növekedésében van. Ezeket a számítógépeket, nevük ellenére, többségében az 1000 főnél többet foglalkoztató cégek veszik. A jóslások szerint, ha ezek a vállalatok maradnak a fő vásárlók, akkor a személyi számítógépek eladása nem fog nőni megfelelő mértékben. A legnagyobb problémát az jelenti, hogy a nagy mennyiségű és sokfajta számítógépet nehéz hálózatba kötni. Ezen a területen sürgető a megfelelő technológia kialakítása.
A miniszámítógép-gyártók különlegesen nehéz helyzetben vannak, mert az IBM PC AT és a Compaq AT kompatibilis gépe közvetlenül veszik fel velük a versenyt. A Data General nevű cég például kénytelen volt 1300 munkatársát elbocsátani és a cég történetében először veszteséggel zárta a félévet. Maga az IBM is arról számol be, hogy közepes méretű számítógépeinek eladása visszaesett. Minden amerikai számítógép-gyártó a beruházások visszaesése miatt szenved. Ilyen helyzetben az IBM is veszélyesebbé válik és erőteljesebben küzd versenytársai ellen. Az iroda-automatizálás területén tett lépései máris komoly problémákat okoznak a Wang számítógép-gyártó cégnek. A Wang 1600 embert volt kénytelen elbocsátani.
A mikroszámítógép-gyártók közül az Apple küzd a legnagyobb gondokkal.
Három telephelyét lezárta és 1200 embert tettek ki állásából. Az Apple hagyományosan a kisebb cégek és az oktatási intézmények között talált vevőkre eddig. A számítástechnikai piacot a felhasználóhoz közelebb szoftver és a hálózatok technológiájának áttörése mentheti ki jelenlegi nehéz helyzetéből.

A szerkesztő azért van, hogy a lap olyan legyen, amilyenek az olvasói!




```
0010 //
0020 // SPRITE BEMUTATO
0030 //
0033 BACKGROUND 0
0034 BORDER 0
0040 DIM ADAT$ OF 63
0042 DIM-X(63)
0050 FOR A:=1 TO 63 DO
0060  READ X(A)
0080  ADAT$:=ADAT$+CHR$(X(A))
0090 ENDFOR A
0091 //
0095 // A 0. SPRITE-BA TOLTJUK AZ ADATOKAT
0096 //
0100 DEFINE 0,ADAT$+CHR$(0)
0110 //
0120 // 4 SPRTE-BA VISSZUK AT A 0.-BAN LEVO ADATOKAT
0123 //
0124 SETGRAPHIC 0
0140 IDENTIFY 0,0
0150 IDENTIFY 1,0
0160 IDENTIFY 2,0
0170 IDENTIFY 3,0
0180 //
0190 // SZIN BEALLITAS
0200 //
0210 SPRITECOLOR 0,2
0230 SPRITECOLOR 1;1
0240 SPRITECOLOR 2,5
0250 SPRITECOLOR 3,8
0260 //
0270 // A MERETEK BEALLITASA
0280 //
0300 SPRITESIZE 0,1,1
0310 SPRITESIZE 1,1,1
0320 SPRITESIZE 2,1,1
0330 SPRITESIZE 3,1,1
0340 //
0350 // A LATVANY
0360 //
0370 SETGRAPHIC 0
0390 A:=0
0391 REPEAT
0392  A:=A+1
0393  MOVETO 125,A*2+20
0395  SETXY 125,A*2+20
0420  SPRITEPOS 0,40,A*2
0430  SPRITEPOS 1,80,A*2
0440  SPRITEPOS 2,120,A*2
0450  SPRITEPOS 3;160,A*2
0460 UNTIL A=100
0480 REPEAT
0490 UNTIL KEY$<>CHR$(0)
0500 END
1000 DATA 0,127,0,1,255,192,3,255,224,3,231,224
1010 DATA 7,217,240,7,223,240,7,217,240,3,231,224
1020 DATA 3,255,224,3,255,224,2,255,160,1,127,64
1030 DATA 1,62,64,0,156,128,0,156,128,0,73,0,0,73,0,0
1040 DATA 62,0.0,62,0,0,62,0,0.28,0
```

A nagyfelbontású grafikáról szóló cikk után, most a COMAL utasítás definiáló képességéről és a sprite grafikát segítő utasításokról olvashatnak.

Ha Basic-ben egy műveletsorozatot többször el szeretnénk érni, akkor, szubrutinokat készítünk. A COMAL-ban ez nem áll rendelkezésünkre. A szubrutinok helyett egy sokkal kényelmesebb és áttekinthetőbb módszert kínál a rendszer. Ez az utasítás a definiálás. Egy általunk meghatározott utasításra a programban bárhol és bármikor hivatkozhatunk, esetleg a program megállítása után parancsként is használhatjuk. Az új üzenetet a gép csak addig tudja felhasználni és visszaemlékezni rá, amíg egy változó az értékét megtartja. Az utasítást nem tudjuk felhasználni többet, pl.: NEW után, RUN után, LIST után stb. Az új parancsot a programból tudjuk csak definiálni. Az utasítás definiálására példának nézzük meg a „TÉGLALAP" című programot! A TÉGLALAP utasítás definiálása a 10. sortól a 80-as sorig tart. Az új művelet nevét a művelet előtt kell megadni, a PROC utasítás mögött. Az általunk készített utasítás neve után zárójelben kell feltüntetni az esetleges paraméterek jelölését (vesszővel elválasztva). Majd azt a sort, amelyben az új utasítás neve található, követi az utasításnak eleget tevő programocska (alprogram). Az alprogram végén a definiált utasítást le kell zárnunk (hasonló módon, mint a ciklusokat) egy ENDPROC-kal, ami után meg kell adni az új parancs nevét. A példaprogramunkban a 100 és 120 közötti sorok használják fel az új utasítást. Egy programban több definiált utasítás is lehet, de egymásba skatulyázni nem tudjuk őket a definiálásnál.

A TÉGLALAP és a HÁZ című programok mutatják be, hogy hogyan is lehet élni ezzel a lehetőséggel.

dore
KA

```
0010 PROC TEGLALAP(HOSSZUSAG,SZELESSEG)
0020  FOR J:=1 TO 2 DO
0030   FORWARD HOSSZUSAG
0040   RIGHT 90
0050   FORWARD SZELESSEG
0060   RIGHT 90
0070  ENDFOR J
0080 ENDPROC TEGLALAP
0085 SETGRAPHIC 0
0090 BACK 70
0100 FOR T:=0 TO 90 STEP 3 DO
0105  TEGLALAP(90-T/2,90-T)
0110  LEFT 5
0115  FORWARD 3
0120 ENDFOR T
```

Sprite kezelés:

A COMAL ugyanúgy, mint a többi grafikai programnyelv lehetőséget nyújt sprite-kezelésre is. A sprite felhasználás első lépése itt is az alakzat megtervezése. A Basic-hez hasonlóan a sprite adatait DATA sorokban helyezzük el, csak a betöltésük tér el egy kicsit a megszokottól. Első lépésként egy 63 karakternyi stringet kell dimenzionálni, ugyanis a COMAL sajnos nem tudja úgy kezelni a szöveges változókat, mint a Basic (minden rendszernek van hibája). A string dimenzionálása a 40. sorban történik. Az adatok tömbbe való beolvasása azért történik, hogy a két DIM közti különbség észrevehető legyen. A sprite-kezeléshez szükséges utasításokat az 1. táblázat tartalmazza.

```
0010 //
0020 // HAZ UTASÍTÁS ES RAJZOLÁS
0030 //
0040 PROC HAZ(MT)
0050  FORWARD MT
0060  LEFT 90
0070  FORWARD MT
0080  LEFT 90
0090  FORWARD MT
0100  LEFT 90
0110  FORWARD MT
0120  LEFT 90
0130  FORWARD MT
0140  LEFT 30
0150  FORWARD MT
0160  LEFT 120
0170  FORWARD MT
0180  LEFT 30
0190  FORWARD MT
0200  LEFT 90
0210  FORWARD (MT/2)-(MT/2)/4
0220  LEFT 90
0230  FORWARD MT/2
0240  RIGHT 90
0250  FORWARD MT/4
0260  RIGHT 90
0270  FORWARD MT/2
0280  LEFT 90
0290  FORWARD (MT/2)-(MT/2)/4
0300  LEFT 90
0310 ENDPROC HAZ
0320 SETGRAPHIC 0
0330 BORDER 0
0340 BACKGROUND 0
0345 PRINT "J"
0350 PENUP
0360 MOVETO 10,50
0370 PENDOWN
0380 FOR A:=1 TO 8 DO
0390  PENCOLOR A
0400  HAZ(A*10)
0410  RIGHT 90
0420  FORWARD (A*10)+10
0430  LEFT 90
0440 ENDFOR A
0450 END
```

DATACOLLISION n, (TRUE v. FALSE)	**Sprite és karakter ütközését vizsgálja, ahol n a sprite száma. A második paraméter TRUE, ha igaz, vagy FALSE ha nem.**
SPRITECOLLISION n, (TRUE v. FALSE)	**Sprite-sprite ütközést vizsgál.**
DEFINE n, xS	**A sprite definiálása. n a sprite száma. x$ az adatokat tartalmazó string változó.**
IDENTIFY n, d	**Az n. sprite-ot azonossá teszi d.-kel.**
PRIORITY n, p	**A prioritás (elsőbbség) beállítása. n a sprite száma, p=0 grafika fölött van a sprite, p=1 alatta.**
SPRITEBACK s1, s2	**Multicolor sprite színei.**
HIDESPRITE n	**n. sprite eltűnik.**
SHOWSPRITE n	**n. sprite megjelenik.**
SPRITECOLOR n, c	**n. sprite színe c. lesz.**
SPRITEPOSITION n, x, y	**n. sprite x, y kordinátába kerül.**
SPRITESIZE n, x, y	**n. sprite x, y irányú nagyítását adja meg.**



GÉPI KÓD III.

Ha egy gépi kódú programot kezdünk írni, legelső teendőnk meghatározni, hogy hol helyezzük el a memóriában. Legcélszerűbb, a felhasználható BASIC-területből lecsípni egy darabot, azt letiltani az interpreter számára, nehogy véletlenül ráírjon valamit, s gépi kódú programunkat oda betölteni, használva a POKE utasítást.

A felhasználható BASIC-terület bővítő nélkül a 4096-os címtől a 7680-as címig terjed. Az interpreter a BASIC-programot a 4096-os címtől kezdi szépen soronként elhelyezni a memóriába. Ha tehát van egy 200 byte hosszú gépi kódú programunk, akkor a memória végét megszabhatjuk 7680 helyett például 7400-ban, amelynek kétbyte-os formája: 232,28 (mivel 28∗256+232 = 7400). A letiltáshoz a következő utasítást kell beadnunk:

POKE 55,232:POKE 56,28:CLR

Ettől fogva az interpreter nem érinti a 7400–7680-ig terjedő területet, ha mondjuk változókat akar elhelyezni a memóriában.

Továbbá felhasználhatjuk gépi kódú rutinok számára a 828–1023-ig terjedő kazettaterülete is, ha nem használjuk a kazettás magnót, vagy a 673–767-ig terjedő memóriaterületet, amit az interpreter ritkán szokott használni. A most következő rövid gépi kódú programot, amit példa-programnak szánok, a 673-as címtől fogjuk a memóriában elhelyezni, így valamenynyi BASIC-területünk szabad marad, és amíg a gépet kikapcsoljuk, addig a program működni fog. A NEW utasításról jó ha tudjuk, hogy egyetlen esetben sem törli a gépi kódú programunkat, így azok egy új BASIC-program megírásakor is felhasználhatók. Az alábbi program a PLOT és COLOR utasításokat fogja megvalósítani a VC–20 gépünkön a@ és a £ szimbólumok használatával, a korábban ismertetett kódok használatának a bemutatására. A programot két részből állítjuk össze. Az első értésére adja az interpreternek, hogy hol kezdődik a második rutinunk; a második végzi a PLOT és COLOR utasítások végrehajtását.

A 776–777 cím arra a ROM-rutinra mutat, amelyik a BASIC-utasításokat kódjaik alapján szétválogatja és az utasítást végrehajtó szubrutint meghívja. A mi programunk közbe lesz iktatva ebbe a folyamatba. A 776–777 cím átírása után a mi kezünkbe kerül előszörre az utasításvizsgálat sora. Megvizsgáljuk, hogy a vagy következik-e, ha nem, akkor visszaadjuk a szót az interpreternek, ha igen, akkor a mi megfelelő szubrutinunk kerül végrehajtásra és csak utána adjuk vissza a szót az interpreternek.

673: LDA 172 = 169, 172	; az akkumulátorba töltjük a 684 cím első felét
675: STA C,776 = 141,8,3,	; és betöltjük a rendszermutatóba
678: LDA 2 = 169,2	; a 684 cím második felét töltjük be
680: STA C,777 = 141,9,3	; és betöltjük a rendszermutatóba
683: RTS = 96	; visszatérés szubrutinból
684: JSR C,115 = 32,115,0	; meghívjuk azt a rutint, amely elhozza a végrehajtás alatt álló sor következő byte-jának a tartalmát
687: CMP 64 = 201,64	; megvizsgáljuk egyenlő-e a@kódjával
689: BEQ PLOT = 240,27	; ha igen, ugrás a PLOT végrehajtó rutinhoz.

Jelen esetben, ha teljesül a feltétel, az ugrási cím a következőképpen számítódik ki
PC = 691 :a = 27
(A PC mindig a soron következő utasításra mutat)
CIM = PC+a+256∗(a>127) = 691+27+256∗0 = 718

691: CMP 92 = 201, 92	; megvizsgáljuk, most egyenlő-e £ kódjával
693: BEQ COLOR = 240,6	; ha igen, ugrás a COLOR rutinra.
695: JSR C,121 = 32,131,0	;ez azonos a 115-című rutinnal, csak nem a következő, hanem az előbb elhozott byte tartalmát hozza el ismét
698: JMP C,51175 = 76,231,199	; mivel sem a @ sem £ nem volt, visszaadjuk a szót az interpreternek
COLOR = 701:JSR C,55195 = 32,155,215	; meghívunk egy ROM-rutint, amely kiszámítja a £ után következő max. 255-ös értéket, és betölti az X regiszterbe
704: STX,C 646 = 142,134,2	; beállítjuk a „karakter színe" mutatót
707: BEQ END = 240,6	; az előbb használt ROM-rutin az akkumulátorba töltötte a szám után lévő byte tartalmát. Ha az kettőspont vagy nulla, akkor Z–bit = 1, tehát az utasításunk befejeződött, és ugrunk az END rutinunkra
709: JSRC, 57853 = 32,253,225	; ha nincs vége az utasításunknak, akkor ez a meghívott rutin ellenőrzi, hogy vessző következik-e, és ha szintaktikusan helyes, akkor kiszámítja a vessző utáni kifejezés értékét és betölti az X-regiszterbe
712: STX C, 36879 = 142,15,144	; ezzel az értékkel beállítjuk a háttér és keretszín mutatót.
END = 715: JMP C, 51118 = 76,174,199	; és visszaadjuk a szót az interpreternek
PLOT = 718:JSRC, 55195 = 32,155,215	; X-regiszterbe betöltjük a @ utáni szám értékét, ez lesz a sorkoordináta
721: CPX,23 = 224,23	; megvizsgáljuk, kisebb-e 23-nál, mivel csak 0–22 sorunk van
723: BCC OK = 144,3	; ha igen, akkor folytatjuk a programot
HIBA = 725:JMP C,53832 = 76,72,210	; ha nem, akkor az Illegol Quantity hibaüzenet-jelzésnek adjuk át a szót.
OK = 728:STX Z,176 = 134,176	; eltesszük sorkoordinátánk értékét egy zérólapos címre

```
730:JSR C,
    57853 = 32,253,225   ; ez a rutin hozza el a másik adatot
                           az X-regiszterben, és megvizsgál-
                           ja, hogy szintaktikusan helyesen
                           vesszővel vannak-e elválasztva
                           egymástól
733:CPX,22 = 224,22      ; megvizsgáljuk, hogy oszlopkoor-
                           dinátánk kisebb-e 22-nél
735:BCS HIBA = 176,244   ; ha nem kisebb, ugrás vissza a hiba-
                           üzenet-rutinra
```

(Itt láthatjuk azt a példát, amikor a > 127 érték áll az utasítás után. Mivel PC = 737:a = 244, ezért CÍM = PC+a+256∗(a>127) = 737+244+256∗(−1) = 737+244−256 = 725)

```
737: TXA = 138           ; az oszlopkoordinátát cserével
738: TAY = 168           ; áttöltjük a Y-regiszterbe
739: LDX Z,176=166,176   ; visszatöltjük X-regiszterbe a sor-
                           koordinátát a zérólapról
741: JSR C,
    58634 = 32,10,229    ; meghívunk egy kurzorállító ROM-
                           rutint
744: JMP
    C,5118 = 76,174,199  ; és visszaadjuk a szót az interpre-
                           ternek
```

Van tehát két működő utasításunk, amellyel tudjuk állítani a kurzor koordinátákat, a karakterszíneket, a háttér és keretszíneket rendkívül gyorsan és röviden, a következőképpen:

@ X,Y : £S1,S2:PRINT"A"

Ahol X a sorkoordináta, Y az oszlopkoordináta, S1 a karakterszín értéke, S2 a háttér és keretszín POKE-értéke. Mindegyik értéket tetszőleges aritmetikai és logikai kifejezésekkel is megadhatjuk. Az IF...THEN utasítás használata esetén a THEN és a használt szimbólumok közé tegyünk kettőspontot pl.:

IF RND(1) > 0,5THEN:@10,10:PRINT,,OK"

Miután elkészítettük a gépi kódú programunkat, töltsük be a gépbe. Erre szolgál ez a következő BASIC-rutin.

```
 10 DATA 169,172,141,8,3,169,2,141,9
 20 DATA 3,96,32,115,0,201,64,240
 30 DATA 27,201,92,240,6,32,121,0,76
 40 DATA 231,199,32,155,215,142,134,2
 50 DATA 240,6,32,253,225,142,15,144,76
 60 DATA 174,199,32,155,215,224,23,144
 70 DATA 3,76,72,210,134,176,32,253
 80 DATA 225,224,22,176,244,138,168,166
 90 DATA 176,32,10,229,76,174,199
100 A = 673
110 READ X
120 POKE A,X : A = A+1:N = N+X
130 IF A < 747 THEN 110
140 IF N <> 8984 THEN PRINT
    "HIBA AZ ADATBEVITELBEN!" END
150 SYS 673: NEW
```

A DATA-ban tárolt gépi kódú programot a 110–130 sorok töltik be a 673-as címtől kezdődően. A 140-es sorban egy adatbeviteli ellenőrzés történik. A 150-es sorban kapcsoljuk be programunkat az INTERPRETER működésébe, majd kitörli a töltőprogramunkat. Az alábbi kis programmal kipróbálhatjuk új utasításainkat

```
10 PRINT CHR$ (147)
20 FOR I = 0 TO 20
30 FOR J = 0 TO 20
40 @I, J
50 £ J
60 PRINT CHR $ (64+J)
70 NEXT : £ 0,60+I :NEXT
80 RUN
```

Tóth Kornél 1082 Bp. Leonardo da Vinci u. 29. fsz. 8.

SZUBRUTINOK

Tisztelt szerkesztőség!

A Commodore VC–20 Basic-je a C–64-hez hasonlóan sajnos nem túl jó. Így adódhat olyan eset, amikor olyan utasításra lenne szükségünk, melyet a CBM Basic nem ismer, vagy másképpen ismer. Ilyenkor néhány ügyes bűvészmutatvánnyal esetleg lehet eredményt elérni, de nem mindig. Ezért közreadok néhány gépi kódú szubrutint, amely megkönnyítheti a programozó munkáját. A programok alapgépen is, és bármilyen bővítéssel kiegészített (3 K, 8 K 16 K) gépen is használhatók. A betöltő program beírása és lefuttatása után a szubrutinok a kikapcsolásig bent maradnak, a Basic-et nem zavarják.

A szubrutinok ismertetése:

1. Kiszámított GOTO utasítás:

Hívása: SYS nnnnn,X ahol nnnnn a szubrutin kezdőcíme (a betöltő program mindegyik szubrutinét kiírja, és amely kezdőcím a bővítésektől függően más és más lehet.)

A szubrutin működése a Basic GOTO utasításával megegyezik, de X nem csak számkonstans, hanem tetszőleges numerikus változó vagy kifejezés is lehet. Ezenkívül C-nek nem kell egésznek lenni, a program az egész részét is veszi. (A SPECTRUM például ismer ilyen utasítást.)

2. Kiírás a képernyő tetszőleges helyére:

A Commodore gépeken a képernyőn a cursor mozgatása programból kissé nehézkes és sok helyet is foglal. Az itt következő utasítás a kiírás vezérlését leegyszerűsíti.

Hívása: SYS nnnnn,X,Y,SZS

Kiírja képernyőre az X,Y koordinátáktól kezdődően SZS-et (SZS természetesen nem csak string lehet, hanem bármi, ami a PRINT utasítás után lehet).

0 X 21 a vízszintes koordináta
0 Y 22 a függőleges koordináta
X és Y pedig tetszőleges kifejezés lehet!

3. Kiszámított RESTORE:

Sajnos a CBM BASIC RESTORE utasítása a data-mutatót csak a program elejére tudja állítani. A SYS nnnnn,X viszont kiszámítja X értékét, és ha van ilyen sorszámú sor, akkor a data-mutatót ennek elejére állítja (a Spectrum ilyet szintén nem ismer!).

4. RE-NEW:

A NEW parancsot hatástalanítja! A program így újra futtatható. Hívása: SYS nnnnn. Vigyázat: ha a NEW után egy változónak értéket adunk, akkor ezt a szubrutint ne használjuk! Tehát csakis program „megnyuvasztása" után használható!

Az igazsághoz tartozik az, hogy a tetszőleges helyre való kiírás ötletét Lángos István könyvéből vettem. Ott ez a szubrutin C–64-re volt meg, én csak átírtam VC–20-ra, valamint kiegészítettem a koordináták vizsgálatával.

Pintér Károly Székesfehérvár, Jáky József Szakközépiskola

Várjuk további ötleteit, ígért szubrutinjait!

```
5 CLR:POKE 36879,25
10 A=PEEK(55)+256*PEEK(56)
20 A=A-118:D=0
30 POKE 56,INT(A/256)
40 POKE 55,A-INT(A/256)*256:CLR
45 TI$="000000"
50 PRINT"▒▒▒▒▒▒▒▒▒ VA▒▒,▒RJ";7-VAL(TI$);"MA▒▒,▒SODPERCET!"
60 A=PEEK(55)+256*PEEK(56)
65 FOR A=A+2 TO A+115
70 READ C$:GOSUB 200:POKE A,C:PRINT"▒▒▒▒▒▒▒▒▒▒▒▒▒▒";7-VAL(TI$)
75 D=D+C:NEXT
80 IF D=14663 THEN 90
84 POKE 56,INT((A-1)/256)
86 POKE 55,A-INT((A-1)/256)*256-1:CLR:PRINT"▒▒▒▒▒▒ADAT HIBA!!"
88 STOP
90 PRINT"▒▒▒  SZUBRUTINOK HIVA▒▒,▒SI▒        CIMEI:▒"
100 PRINT"▒▒  KISZA▒▒,▒MITOTT GOTO:▒ ▒      SYS";
105 PRINTPEEK(55)+256*PEEK(56)+2;"▒,X"
110 PRINT"▒▒  KIIRA▒▒,▒S TETSZO▒▒▒LEGES▒        HELYRE:▒"
115 PRINT"▒   SYS";PEEK(55)+256*PEEK(56)+15;"▒,X,Y,SZ$▒"
120 PRINT"▒ KISZA▒▒,▒MITOTT RESTORE:▒▒    SYS";
122 PRINTPEEK(55)+256*PEEK(56)+55;"▒,X"
125 PRINT"▒▒ RE-NEW:▒SYS";PEEK(55)+256*PEEK(56)+89
130 GET R$:IF R$="" THEN 130
140 NEW.
190 END
200 L$=LEFT$(C$,1)
210 R$=RIGHT$(C$,1)
220 L=VAL(L$)
230 IF L=0 AND L$<>"0" THEN L=ASC(L$)-55
240 R=VAL(R$)
250 IF R=0 AND R$<>"0" THEN R=ASC(R$)-55
260 C=16*L+R:RETURN
1000 DATA 20,FD,CE,20,8A,CD,20,F7
1010 DATA D7,4C,A3,C8,EA,20,FD,CE
1020 DATA 20,9E,D7,E0,16,B0,18,8A
1030 DATA 48,20,FD,CE,20,9E,D7,E0
1040 DATA 17,B0,0C,68,A8,20,F0,FF
1050 DATA EA,20,FD,CE,4C,A0,CA,A2
1060 DATA 0E,4C,37,C4,EA,20,FD,CE
1070 DATA 20,8A,CD,20,F7,D7,20,13
1080 DATA C6,B0,05,A2,11,4C,37,C4
1090 DATA A5,5F,A4,60,E9,01,B0,01
1100 DATA 88,85,41,84,42,60,EA,A0
1110 DATA 01,98,91,2B,20,33,C5,A5
1120 DATA 22,A4,23,18,69,02,90,01
1130 DATA C8,85,2D,84,2E,20,60,C6
1140 DATA 4C,74,C4,EA

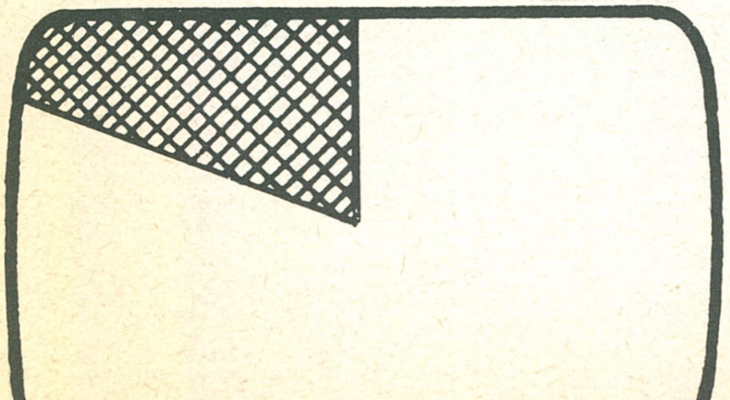
READY.
```

MICOLOR NYERŐ



Ha programírási feladatot adunk tisztelt olvasóinknak, akkor az mindenképpen kötelez bennünket arra, hogy a beérkezett munkák közül legalább egyet, a legjobbat vagy legtanulságosabbat közöljük. MICOLOR nyerő pályázatunkban sok iskolai szakkör vett részt, sok jó és rossz programot küldött be. Ezúttal az 1. számú feladatról ejtünk néhány szót, s közöljük az egyik programot is.
A feladat kiírása a következő volt:

Készítsünk kaleidoszkópot! Tehát egy grafikai programot kérünk, a következő megkötésekkel:
1. A program HT gépre készüljön BASIC nyelven, most gépi kódú rutinok használata nem engedélyezett!
2. A képernyő egyik nyolcadában (l. az ábrát!) minden állapotban legyen egy véletlenszerű minta, a többi nyolcadot ennek tükörképeivel töltse ki a gép.
3. Az állapotok ne úgy váltsák egymást, hogy először a „vezérnyolcad" változik, utána többi, hanem szép, szimmetrikus módon történjen a változás.
Jó szórakozást!



A feladatra 40 megoldás érkezett be. Gondot általában az okozott, hogy a megadott téglalap-nyolcadot tengelyesen tükrözgetve nem lehet lefedni a képernyőt. Néhány jellemző kikerülési mód (értékességi sorrendben, először a jobbak):

- a háromszöget a középső függőleges tengely irányában kicsinyítsük egyenlő szárúvá, tükrözzük az átfogóra, majd nyújtsuk vissza a bal oldal felé (a kicsinyítés-nyújtást csak az x koordinátán kell elvégezni)!
- a háromszöget tükrözzük középpontosan átfogója felezőpontjára!
- téglalap helyett használjunk négyzetet vagy kört!
- téglalapot használjunk, de csak a közepén benne foglalt négyzetben végezzünk tengelyes tükrözést az átlóra!
- stb.

Szépséghibának vettük, ha a program a teljes képernyőt használja, de nem dupla pontokkal (mert az erősen téglalap alakú pontok csúnyák, s a szimmetriát is rontják); s levontunk pontot a lassúságért is.
Mint kiderült, nem sokan látták még kaleidoszkópot, ez kissé rányomta a bélyegét a megoldásokra; igazi kaleidoszkópszerű programot nem kaptunk; általában egy állapot beállta után minden lépésben 1 pont és a 7 tükörképe változott. Jobban örültünk volna olyan megoldásoknak, ahol egyszerre nagyobb változás történik, megfelelő sebességgel például a képernyőn felülről lefelé. Ehhez az értelmezéshez leginkább a Piarista Gimnázium programjában találtunk hasonlót, most mégis a Berzsenyi Dániel Gimnázium programját közöljük, mert a téglalap alakú kaleidoszkópok közül az övék volt a legszebb.

Kaleidoszkóp program – Bp. Berzsenyi Dániel Gimnázium 3. haladó szakkör

A program egy olyan kaleidoszkópot modellez, amelybe két, egymással 45°-os szöget bezáró tükör van.
A kaleidoszkóp alakja 3-féle lehet:

- kör
- négyzet
- téglalap (a négyzet „elnyújtott" változata)

A program elindítása után (RUN) a három változat közül kiválasztjuk a megfelelőt (A/B/C). Ezután a gép a billentyűzettől függetlenül folytatja a programot (kivétel: BREAK).

Az egyes programrészek funkciói:
110: Kisbetűk használatát engedélyezi.
140–230: Üzemmód-választás.
240–310: a) és b) módban keretrajzolás.
320–370: A kép változtatása.
380–440: c) módhoz kör rajzolása.
450–630: b) és c) módhoz szubrutin.
640–850: a) módhoz grafikai szubrutin.

```
10 REM ***********************
20 REM * KALEIDOSZKOP        *
30 REM * BERZSENYI D. GIMN. *
40 REM * 3. HALADO SZAKKOR  *
50 REM ***********************
100 DEFINT A-S,U-Z
110 POKE 16414,71:POKE 16415,49
120 RANDOM:RANDOM:RANDOM
130 CLS
140 PRINT "KALEIDOSZKOP"
150 PRINT
160 PRINT "MILYEN GRAFIKAT HASZNALJON
A PROGRAM?"
170 PRINT
180 PRINT "a) 61x45"
190 PRINT "b) 45x45"
200 PRINT "c) 22 SUGARU KOR"
210 IF INKEY$<>"" THEN 210
220 A$=INKEY$
230 IF A$="A" THEN GR=1 ELSE
   IF A$="B" THEN GR=2 ELSE
   IF A$="C" THEN GR=3:GOTO 380
   ELSE 220
240 IF GR=2 THEN A=9:B=55 ELSE
   A=1:B=63
250 CLS
260 FOR I=A TO B
270 SET(I,0):SET(I,46)
280 NEXT I
290 FOR I=0 TO 46
300 SET(A,I):SET(B,I)
310 NEXT I
320 X=RND(45)-23
330 Y=RND(45)-23
340 ON GR GOSUB 640,450,360
350 GOTO 320
360 IF X*X+Y*Y>484 THEN RETURN
370 GOTO 450
380 T=0:T0=.02
390 CLS
400 X=INT(46*T/(1+T*T)+.5):Y=INT(23*(1-T*T)/(1+T*T)+.5)
410 T=(T+T0)/(1-T*T0)
420 GOSUB 460
430 IF T<=.42 THEN 400
440 GOTO 320
450 IF POINT(32+X,23+Y) THEN 550
460 SET(32+X,23+Y)
470 SET(32+X,23-Y)
480 SET(32-X,23+Y)
490 SET(32-X,23-Y)
500 SET(32+Y,23+X)
510 SET(32+Y,23-X)
520 SET(32-Y,23+X)
530 SET(32-Y,23-X)
540 RETURN
550 RESET(32+X,23+Y)
560 RESET(32+X,23-Y)
570 RESET(32-X,23+Y)
580 RESET(32-X,23-Y)
590 RESET(32+Y,23+X)
600 RESET(32+Y,23-X)
610 RESET(32-Y,23+X)
620 RESET(32-Y,23-X)
630 RETURN
640 X=ABS(X):Y=ABS(Y)
650 N=0
660 GOSUB 680
670 Z=X:X=Y:Y=Z
680 Z=INT(X/3):M=X-3*Z
690 IF M=2 THEN M=3
700 X1=4*Z+M
710 GOSUB 730
720 IF M=1 THEN X1=X1+1 ELSE RETURN
730 N=N+1
740 IF N=1 THEN PP=POINT(X1,Y)
750 IF PP THEN 810
760 SET(32+X1,23+Y)
770 SET(32+X1,23-Y)
780 SET(32-X1,23+Y)
790 SET(32-X1,23-Y)
800 RETURN
810 RESET(32+X1,23+Y)
820 RESET(32+X1,23-Y)
830 RESET(32-X1,23+Y)
840 RESET(32-X1,23-Y)
850 RETURN
```



Múlt havi cikkünkre egy azonnali reagálás érkezett. Egy beszámoló. Tapasztalatátadásra buzdítottunk, így hát tartva szavunkat, máris közöljük e beszámolót.

Az NJSZT Házikészítésű Computer Clubon belül működik az APPLE szekció is, amelyet Európa egyik legnagyobb komputerklubja, az APPLE USER GROUP EUROPE e.V. magyarországi regionális csoportjaként ismerte el. Az A.U.G.E.-nek kb. 5500 bejegyzett tagja van elsősorban német nyelvterületen. A magyar csoportnak 50 tagja van, amelyből 30 rendszeresen vesz részt a klubfoglalkozásokon. Az önálló egyesület regonális csoportokra és munkacsoportokra oszlik. A regionális csoportok rendszeres találkozót tartanak, amelyeken mindenki, akinek van bárminemű problémája, segítséget kaphat. A munkacsoportok szakirányú tevékenységet végeznek, mint pl. a PASCAL, CP/M, játék, távadatátvitel, mesterséges intelligencia, iskolai adminisztráció, potprocesszorok, adatbáziskezelés, könyvtár kezelése és irodalom, hogy csak a legfontosabbakat, említsem a több mint húsz munkacsoportból. A magyarországi regionális csoport – most már hagyományként – szervezi nyaranta a nemzetközi APPLE-tábort, amelyre kéthetes időtartamra – elsősorban Nyugat-Európából – várunk vendégeket. Az eddig Miskolcon, Szentendrén és Budapesten rendezett táborokon voltak vendégek NSZK-ból, Ausztriából, Olaszországból, Hollandiából és természetesen Magyarországról.

A táborokon szemináriumokat tartunk a fenti munkaközösségi témákban. A legnépesebb szekció mindig a PASCAL, úgyhogy a táborok alkalmával több PASCAL-szekciót is kellett alakítani: PASCAL kezdőknek, PASCAL angol nyelven és PASCAL haladóknak, ahol a haladók mindig workshop formában dolgoztak ki egy nagyobb programrendszert, amely később klubszoftverként kerül terjesztésre a klubtagok között. További szekciók voltak BASIC kezdőknek, 6502-es és 68000-es ASSEMBLER, CP/M és dBASE, valamint hardver. A szemináriumokat mindig olyan szakmailag elismert egyéniségek vezetik, akik az A.U.G.E.-n belül és ezen kívül is már nevet szereztek. Ők eddig egy kivétellel Nyugat-Európából jöttek.

A szemináriumvezetők és a hallgatók is magukkal hozzák saját APPLE számítógépüket. Ennek behozatala és visszavitele eddig – egy eset kivételével – nem okozott gondot a vámhatóságoknak, mivel az NJSZT titkárság olyan kezességet vállaló levelet küldött minden résztvevőnek, mely alapján a berendezések vámkezelése megtörtént.

A szemináriumvezetők nem fizetnek részvételi díjat, mivel ezt a szemináriumokon résztvevők fizetik ki.

De ha már ide jöttek a vendégek Magyarországra, természetesen mutatunk is valamit hazánkból. Így kirándulásokat szerveztünk legszebb tájainkra, így Egerbe, Visegrádra, Szentendrére, Hortobágyra, Hollókőre, Pécsre és Budapestre.

A résztvevők általában azzal búcsúznak, hogy ,,Jövőre újból jövünk!''. Ezt alátámasztja az résztvevők emelkedő száma is. Jövőre augusztus második felében szeretnénk a tábort Veszprémben vagy környékén rendezni. Az előzetes program szerint ismét három PASCAL-szekció, ASSEMBLER és CP/M szekció lesz. Újdonság lesz a LISP, mint a mesterséges intelligencia egyik nyelve.

Eddig mindig a megfelelő helyszín kiválasztása jelentett problémát, hisz ilyen jellegű tábort csak szép környezetben fekvő, megfelelő felszereltségű üdülőben lehet rendezni, ahol az általános napi étkeztetést és a szemináriumok lebonyolítását is lehet biztosítani.

A legnagyobb nyereség viszont a kölcsönös megismerkedés, segítség és a számítástechnikai ismeretek gyarapítása. Itt amatőrök segítettek amatőröknek, és emellett mindenki jól érezte magát.

A HCC APPLE szekciója egyébként minden hó első és harmadik keddjén találkozik az NJSZT termeiben IBp V., Báthori u. 16.). Az első kedden mindig beszélünk az APPLE újdonságairól, a klub belső életéről és bemutatunk új programokat. A harmadik kedden szemináriumi jelleggel beszélünk általános érdeklődésre számot tartó témákról.

Diebel Dietrich, a HCC APPLE szekciójának vezetője

MARKOV-I-Z-É

Ajánlás:

Amikor kezembe vettem a cikket, kicsit bosszús voltam: matematikus öcsémnek megint „elsült" az agya, pedig én egy szórakoztató programot kértem a Bit-let számára. De aztán amikor másodszor is kihűlt a vacsora, mert képtelen voltam felállni a gép elől, úgy döntöttem, mégiscsak jó lesz ez. (Ehhez hozzátartozik, hogy számítógépes játékkal eddig még félóránál többet nem sikerült eltöltenem.) Tehát senki ne riadjon vissza a definícióktól.

Lukács József

Vágjunk rögtön a közepébe, lássuk a definíciókat.

Tekintsünk egy valamilyen jelkészlet (pl. ASCII kód) elemeiből álló véges hosszú sorozatot. Ezt nevezzük *szónak*. Ha e szó elejéről és/vagy végéről elhagyunk tetszőleges jeleket, akkor egy új szót kapunk (esetleg üres szót). Az új szó az előzőnek *részszava* lesz. Ha egy részszót egy másik jelsorozattal helyettesítünk, akkor ismét egy új szót kapunk. Pl. a GIZIKE szó IZI részszavának ÖZE helyettesítésével új szót kapunk: GÖZEKE. Itt IZI → ÖZE a *helyettesítési szabály*.

Most tekintsünk n darab helyettesítési szabályt meghatározott sorrendben:

$X_1 \to Y_1 \quad X_2 \to Y_2 \ldots X_i \to Y_n$

Alkalmazzuk a szabályokat az alábbiak szerint!

Vegyünk egy szót – ez lesz az *Inputszó*. Próbáljuk meg ezen a szón alkalmazni az első helyettesítési szabályt. Ha nem sikerül (X_1 nem része az Inputnak), akkor vegyük a második szabályt, és így tovább. Ha alkalmazható a szabály, akkor végezzük el a helyettesítést, és az új szóval kezdjük elölről az egészet. További kikötés, hogy ha egy szabály alkalmazható, akkor X részszó balról első előfordulását kell helyettesíteni Y-nal.

Ha már egyik szabály sem alkalmazhato, akkor álljunk meg. Az utolsó szó a végeredmény, az *Outputszó*.

Érdemes felhívni a figyelmet, hogy a szabályok sorrendje nagyon lényeges, valamint hogy a szabályok jobb és bal oldala nem feltétlenül egyenlő hosszú. A jobb oldal lehet akár üres is!

Természetesen előfordulhat, hogy sohasem tudunk leállni. Ekkor a szavak végtelen sorozata keletkezik.

Összefoglalva egy olyan eljáráshoz jutottunk, aminek van egy bemenete, és szolgáltat egy kimenetet, vagy végtelen sorozatba kezd – elszáll.

Tulajdonképpen a szokványos számítógépek ugyanezt csinálják, csak más formában van megadva az algoritmus, vagyis a program. Bizonyítható, hogy minden olyan feladatra, ami más számítógépeken megoldható, lehet az előbbiek szerinti „programot" írni, és megfordítva. A *program* itt persze felcserélési szabályok adott sorrendű összessége.

Az itt leírtakat először A. A. Markov definiálta, így róla *Markov-algoritmusnak* nevezték el.

Példák

Az eddigiek alapján talán még nem világos, hogy a valóságban hogyan is lehet egy Markov-féle elven működő számítógépre programot készíteni. Ennek megértésére legegyszerűbb, ha megtekintünk néhány mintaprogramot. Az Input elejét és végét mindig # jelzi. Első példánk 0 1 és 2 karakterekből álló tetszőleges jelsorozatokat rendez sorba úgy, hogy elöl álljanak a nullák, utána az egyesek, majd a kettesek.

A program így néz ki.

1. 10 → 01
2. 20 → 02
3. 21 → 12

A program futásának menete 101210 Inputszó esetén:

1. #011210# 1. #011201# 2. #011021#

1. #010121# 1. #001121# 3. #001112#

Először az 1. szabályt alkalmazzuk a szó elején, majd újra 1-et a sor végén. A következő körben az 1. nem alkalmazható, viszont a 2. igen- a negyedik pozícióban. Az így keletkezett szóban ismét 1.-et lehet alkalmazni. Az ötödik lépésben megint 1., hatodszorra pedig sem 1. sem 2. hanem csak 3. működik. A hetedik próbálkozásra már egyik szabály sem működik az Outputszó tehát 001112 – a számok rendezve vannak.

Ha itt a szabályokat felcseréljük, a végeredmény nem változik meg, viszont a rendezést más sorrendben végzi el a gép, érdemes kipróbálni!

A második példa Inputja tetszőlegesen sok „!". A program megszámolja (tízes számrendszerben), hány ! van az Inputban.

1. 0: → 1
2. 1: → 2
3. 2: → 3
4. 3: → 4
5. 4: → 5
6. 5: → 6
7. 6: → 7
8. 7: → 8
9. 8: → 9
10. 9: → :0
11. #: → #1
12. /! → :/
13. #! → #1/

Ebben a programban az utasítsok nagy része 1–11-ig a számolás menetét definiálja, míg a 12 egy újabb !-et szüntet meg úgy, hogy egyben elindít egy számolást, végül a 13 indítja el az egész folyamatot. A program futása a mellékelt számítógépes modellben

egyszerűen követhető. Azt viszont érdemes megfigyelni, hogy az újonan bevezetett jeleknek (: és / de lehetne más is) külön jelentésük van. A / választja el a felkiáltójeleket a számlálótól, a : pedig azt jelenti, hogy az előtte álló számot növelni kell. A növelést mint műveletet itt külön definiálni kellett, mert a Markov-automata csak karakterekkel dolgozik, nem tudja, hogy vannak számok, és azoknak sorrendje. Ilyen értelemben a 1–10 utasításokat felfoghatjuk a tízes számrendszer definíciójaként is!

Egyébként itt már az utasítások nem mind cserélhetők fel. Például a 12 nem kerülhet előbbre, viszont a 13 bárhol jó lenne, és 1–11 is tetszőleges sorrendben állhat.

Még egy példa, ahol már csak a programot közöljük. Egy szöveg szavait különválogatja úgy, hogy a fölösleges betűközöket kihagyja, és a szavakat / választja el.

1. #⊥ → #
2. U # →
3. /└┘/
4. → /

Ezen kívül persze még sok mindent szellemesen meg lehet oldani Markov-programokkal. Néhány probléma, amit megoldásra javasolunk.

bináris összeadó Inputja pl. 1101001+1010
binárisból decimálisba oda-vissza fordító
BCD összeadó, szorzó stb.
és még ami kinek-kinek eszébe jut.

Persze bonyolultabb programok írásához érdemes az algoritmus definícióját kiterjeszteni. Pl. egy jolly-joker (legyen a jele ✱) bevezetése igen hasznos lenne. Ezt úgy kell érteni, hogy a helyettesítési szabályban a ✱ bármilyen karaktert jelenthet. Az algoritmus további bővítésének lehetőségeit, és azoknak az interpreter programba való beillesztését szintén az olvasóra bízzuk!

Az interpreter

A Markov-programok megírásához és lefuttatásához jó segédeszköz a cikk mellett leközölt Basic program, ami egy Markov-féle számítógépet modellez. A program eredetileg HOMELAB 3 gépre készült, de könnyen átírható más típusra is.

A$(I) és B$(I) az I-edik szabály bal és jobb oldalát tárolja, a B$ pedig az Inputszó, ill. a belőle származó módosítások.

A programban 10–90-ig történik a menü lekezelése. 120–220-ig a programok szerkesztője – vagyis a Markov interpreter editorja található. (120–160 listázza a programot, 170–220 pedig új sort vesz be.)

300-tól a program futtatója következik. Először az Inputszót veszi be, majd 320–350 megkeresi a legelőször alkalmazható szabályt. 360–370 végrehajtja a helyettesítést, és kiírja a gép aktuális állapotát. Így menet közben követhető, hogy mi történik. 380–400 a billentyűzet figyelése. Itt meg lehet állítani a futtatást, vagy vissza lehet térni. 500 a végeredményt írja ki.

A program a korábban mondottakkal egyezésben az inputszó elé és mögé egy-egy #-et tesz.

A Markov-programok futtatása közben senki ne lepődjön meg a sebességen: egy-egy szabály kitalálása 1–10 másodpercig is eltarthat, hiszen állandóan a különben is lassú Basic leglassúbb részét, a sztringműveleteket használjuk. Gépi nyelven ez nyilván sokkal gyorsabb lenne, de a mai számítógépekben sajnos így sem lenne igazán hatékony ez az algoritmus. Alkalmas hardverrel viszont igen egyszerűen programozható gépet lehetne készíteni. Hatékonyságát az is növelhetné, hogy az egymástól független szabályokat egyszerre is lehetne tesztelni, így a párhuzamos működést is meg lehetne oldani.

Az is nyilvánvaló, hogy ez a nyelv végül is nem alkalmas hosszadalmas aritmetikai műveletek elvégzésére, vagy folyamatszabályozásra. Ellenben szövegszerkesztésre, adatkezelésre, szintaktikai elemzésre, szövegek kódolására kiválóan alkalmas lehetne, hiszen programjai tulajdonképpen nem mások, mint az adott feladat pontos és szigorú definíciói.

Ezért ha a napi gyakorlatban ma még közvetlenül nem is használható – mindenkinek épülésére szolgál megismerkedni egy más elvű gondolkodásmóddal.

Lukács Endre

```
10 L=255: Dim A$(50): Dim B$(50)
20  Print chr$(12)
30  Print  cur 10,0"1. Programozás" cur 10,2"2. Futtatás"
40 A$=Inkey$
50  If A$="1" then  Goto 120
60  If A$="2" then  Goto 300
90  Goto 40
120  Print chr$(12): For V=1 to 50
130  If A$(V)="" then  Pop : Goto 170
140  Print  cur 32*int(V/26),V-25*(V>25);rgh$(str$(V),2)". ";
150  Print A$(V)" --> "B$(V)
160  Next
170  Input  cur 0,26"A parancs sorszáma:"A$
180  If A$="" then  Goto 20
190 V=val(A$): If V>50 then  Goto 20
200  Print : Input "Az első szó:"A$(V)
210  If A$(V)="" then  Goto 120
220  Print : Input "A második szó:"B$(V): Goto 120
300  Print chr$(12): Input  cur 0,5"Az input szó:"B$
306 B$="#"+B$+"#": Print  cur 0,10"Az aktuális szó:"
307  Print  cur 0,15"Az aktuális parancs:"
320  For V=1 to 50:A=len(A$(V)): If A=0 then  Pop : Goto 500
330  For X=1 to len(B$)+1-A
340  If mid$(B$,X,A)=A$(V) then  Goto 360' Next
350  Next : Goto 500
360  Print  cur 0,11;B$,, cur 0,16;A$(V)" --> "B$(V),
370  Pop :B$=lft$(B$,X-1)+B$(V)+mid$(B$,X+A,L)
380 A$=Inkey$: If A$="" then  Goto 320
390 A$=Inkey$: If A$="S" then  Pop : Goto 30
400  If A$=" " then  Goto 320  Goto 390
500  Print  cur 0,20;"Az output szó:": Print B$
520  Goto 30

Ok
```

ötlet

ideák

TÉZISEK A SZÁMÍTÓGÉP ALKALMAZÁSÁRÓL

Pécsi olvasónk, Kiss Tibor írása régebben érkezett szerkesztőségünkbe. Olyan témának sok ágát-bogát igyekszik összegezni, amely bennünket is izgat: hol, mikor, mire és miért használják, vagy éppenséggel az istennek sem használják a számítógépet. Melyik vállalatot milyen meggondolások vezeti, amikor gépet vesznek a cégnél, vagy leszavazzák a gépvásárlást? Kiss Tibor írását gondolatébresztőnek is szánjuk; kíváncsiak vagyunk, mit gondolnak ugyanerről olvasóink.

Szidják a számítógépeket. A kezdeti, feltétel nélküli lelkesedés már alábbhagyott, és az ábrándképeket felváltotta a rideg valóság. Ahol nem volt megfelelően megalapozott a számítógépek alkalmazása, ott sokszor azokat okolják a sikertelenségért. A tanulópénzt azonban mindenhol meg kell fizetni, és ki többet, ki kevesebbet fizet.

A személyi számítógépekkel egy fejlett technológiát képviselő eszköz került a vállalatok kezébe, amely a fejlett országok vállalatainál már komoly sikereket ért el. Azonban a szocialista vállalat más közeg a számítógép számára; az alkalmazás lehetőségeit itt külön meg kell vizsgálni.

A bevezetés feltétele ideális esetben, hogy a haszon nagyobb legyen a ráfordításnál. Ez az árbevétel-költség oldaláról, de egyéb oldalról is mérlegelendő, mint pl. munkafeltételek, az eredmények pontossága, az információk gyorsasága, tehát amely tényezők csak közvetetten fejezhetők ki pénzben.

A vállalatoknál azonban nem mindig ilyen ideális a helyzet. Egyik ilyen eset: a számítógépeken rendkívül bonyolult feladatokat is meg lehet oldani gyorsan, és ez sok ember munkáját teheti fölöslegessé. Ez biztosan kifizetődő a vállalat számára, amennyiben az árbevétel-költség oldalról nézzük a dolgot. Azonban lehet, hogy egyéb okok miatt szüksége van a vállalatnak a létszámra, pl. bérszínvonal, munkaerő-tartalék, szociálpolitikai okok miatt, s így nem érdeke a létszámmegtakarítás. Találhatók még más, a tiszta gazdasági logikától eltérő megfontolások, amelyek azt mondják, hogy mégsem kell a számítógép.

Amennyiben indokolt az alkalmazása, úgy ennek a lehetőségét is meg kell teremteni, tehát létre kell hozni a személyi számítógépek alkalmazásának közvetlen feltételeit. Ezek a számítógépet felhasználókra, magára a gépre és az alkalmazás helyére vonatkoznak, valamint ennek a három tényezőnek együttes megvalósulására, amely meghatározza az alkalmazás hatékonyságát.

Szükséges a megfelelő számú szakember biztosítása a gépek kezelésére, az alkalmazás lehetőségeinek ismertetésére, és legalább ennyire fontos, hogy a vállalat dolgozói is lássák a számítógép illeszkedését a vállalati tevékenységrendszerbe. Amennyiben kisebb, pl. Commodore 64 nagyságrendű gépről van szó, akkor lehetőség szerint biztosítani kell, hogy minden érdeklődő hozzáférjen a géphez tanulmányozni, megismerni és felhasználni azt saját munkájában. Ezzel ugyanis sokkal jobban biztosítható a befogadó közeg azonosulása a feladattal, leküzdhető az esetleges idegenkedés ettől az ismeretlen, néhol misztifikált eszköztől.

A személyi számítógépek sok típusa ismeretes. Ezek osztályozhatók bizonyos paraméterek szerint, mint pl. kapacitás, műveleti sebesség, a csatlakoztatható tárolók, nyomtatók megbízhatósága stb. Így viszonylag egyértelműen hozzárendelhetők a megfelelő gépek a megfelelő feladatokhoz.

Az alkalmazás helyét mindig a funkció, a munkafolyamat jellege határozza meg, amennyiben ez technikailag megvalósítható. Alapvető szempont, hogy azon a területen kell üzemeltetni őket, ahol a vállalat számára – valamilyen meggondolásból – ésszerűek. Ilyen lehet a számítások, adatok pontossága és gyorsasága. Pl. üzleti tárgyalásoknál igen lényeges, hogy egy üzlet milyen nyereségességet ígér. A határidő munkáknál is fontos lehet, pl. valamelyik időszak végén, amikor a részadatok egyszerre érkeznek be, és a végeredmény is lehetőleg azonnal kellene. Az unalmas, monoton számításokat gépesítve emberibb munkafeltételeket teremthetünk.

A három közvetlen feltétel megléte azonban még csak látszatharmónia, mivel a gépek hatékonysága igen nagy mértékben függ attól, hogyan ötvözik össze a három tényezőt, hogyan illesztik bele a vállalati folyamatok rendszerébe.

Rendkívül lényeges a megfelelő előkészítés. Amennyiben elegendő pénz áll a vállalat rendelkezésére, úgy az első feltétel már megoldottnak tekinthető. A harmadik feltétel, az alkalmazás helye határozza meg, hogy milyen gépet érdemes vásárolni, így ezt bízzuk szakértőre. Ha nincs a vállalatnál, akkor külső segítséget kell igénybe venni. Ez még mindig sokkal gazdaságosabb, mint az elhibázott gépvásárlás.

A gép ismerete után a többi előkészítő munkát a vállalat önállóan is meg tudja oldani. A kívánt feladatok végrehajtásához szükséges progra-

moknak milyenek az inputadat-igényeik, valamint az outputadatok milyen csatornákon keresztül kerülnek el az azokat felhasználókhoz. Az ebben a rendszerben felmerülő problémákra (pl. bizonyos helyeken nincs olyan munkaerő, amelyik az új feladat végrehajtására képes lenne) a gépvásárlás előtt kell megoldást találni.
Az előkészítést akkor tekinthetjük befejezettnek, ha látjuk az egész jövőbeni folyamat struktúráját, az adatok keletkezését, géprevitelét, az outputadatok továbbítását és felhasználását. Ezután tudjuk csak megítélni, hogy mennyi hasznot hozhat a számítógép, többletjövedelem vagy költségcsökkentés formájában.
Ezután következhet a közvetlen megvalósítás. Mivel több intézmény, társulás is foglalkozik gépek eladásával, tanfolyamokkal, programkészítéssel, ezért a vállalat számára legelőnyösebb vétel érdekében előzetesen tájékozódni kell az összes lehetőségről.
A gépvásárlás után ugyancsak tájékozódni kell a szoftverárakról, tehát hasonló volumenű feladatok elvégzését általában mennyiért és mennyi idő alatt végzik el. Kell tudni azt is, hogy az adott feladatok elvégzéséhez szükséges gép- és programozási ismeret mennyi idő alatt sajátítható el. Ezek igen fontos tényezők, mivel nem biztosítja semmi azt, hogy akár állami intézmények, akár magántársulások ne kérjenek többszörös árat, vagy ne adjanak három-négyszeres tanfolyamidőt.
Azok, akik majd a program output adatait fogják használni, tudják, hogy milyen adatok, információk kellenek a munkájukhoz. Ez a minimumszint, innen indul el a programkészítés. Az összes inputadat összegyűjtése után tisztázódnak, hogy milyen többletinformációk nyerhetők még különösebb ráfordítás nélkül.
Ezután lehet véglegesíteni az outputlisták tartalmát és formáját. Ez már elvileg végleges formának számít, mivel a programozó ezen nem változtat. Amennyiben rossz volt az előkészítés, úgy könnyen előfordulhat, hogy rossz lesz a később állandóan rendelkezésre álló információ is. Ha pedig a felhasználó végül az íróasztalfiók lesz, úgy különösebb értelme nincs a gépnek.
Vannak lényeges személyi feltételek is. Kell, hogy legyen egy ember a vállalatnál, aki átlátja az egész folyamatot, annak szervezeti és személyi vonatkozásait is. El kell érni, hogy az összes érintett személy értse azt: milyen változás lesz a gép üzembe állítása után a munkakörükben, illetve abban a vállalati folyamatban, amelyben közreműködnek. Ehhez előbb meg kell ismertetni mindenkit – legalább általánosságokban – a számítógéppel. Erre – a közvetlen hozzáférés biztosításán kívül – alkalmasak még a gépkezelési, programozási tanfolyamok; ahol az eredménytelen, ott a közvetlen munkatársak segítségét kell igénybe venni. (Ezek természetesen az anyagi ösztönzés mellett, nem helyett alkalmazandók.)
Nem véletlenül fogalmaztam az előzőekben ilyen általánosan. A személyi számítógépek felhasználásának lehetőségei ugyanis szinte korlátlanok. A működésük, munkamódszerük lényegét kell ahhoz megértenünk, hogy megtaláljuk a vállalatoknál, intézményeknél azokat a területeket, ahova ezt a fajta feladat megoldó készséget éppen be lehet illeszteni. Honnan lehet ezt megérteni?
Egyik legkézenfekvőbb módja az, ha kiindulunk a nyugaton elterjedtnek mondható, s már hazánkban is terjedő programcsomagokból. Ezek ilyen mértékű elterjedése ugyanis csak annak köszönhető, hogy igen jól felhasználhatók a vállalati gyakorlatban.
Három lényeges fejlesztési terület figyelhető meg. Az egyik az, amikor a vélt vagy tényleges változások hatását a számítógép pillanatok alatt kiszámolja. Gondoljunk itt pl. árkalkulációra, ahol egy alapanyag-árváltozás hatásának végigvezetése az azt felhasználó termékek árában csak percek kérdése. Fontos segédeszköz lehet pl. a vállalati vezetés számára a tervezés folyamán; esetleges rossz döntéseket lehet könnyen megelőzni. (L. pl.: Elektronikus feladatlap VU-CALC ZX Spectrumra. Mikroszámítógépes Magazin 1981/2. 21–22. old. Móczó József.)
A másik az, hogy az adatok tömegéből tudja kiemelni a lényegesnek tartott elemeket, tendenciákat. Így sokkal érthetőbbé válnak az azt felhasználók számára. (L. pl.: Üzleti grafika. Mikroszámítógépes Magazin 1984/5. 16. old. Simon Iván.)
A harmadik az adattárolás, nyilvántartás területe, beleértve a különböző levélformák, ügyiratok tárolását. (Különböző szövegszerkesztő programok, file-kezelő rendszerek; l. pl. Magyar Szövegszerkesztés a Spectrumon. Ötlet, 1984. december 20. Szekfű András.)
Természetesen a három részt lehet egymással kombinálni is, hogy végül az alkalmazásnak megfelelő szoftver rendelkezésre álljon. (L. pl.: Személyi számítógép a közgazdasági munkában. Mikroszámítógépes Magazin, 1984/5. 26. old. dr. Bódis Béla.)
Végül ki kell hangsúlyozni, hogy ami megvan a fejekben, azt rá lehet tenni számítógépre, de ami nincs, annak megoldását nem lehet várni a számítógéptől.

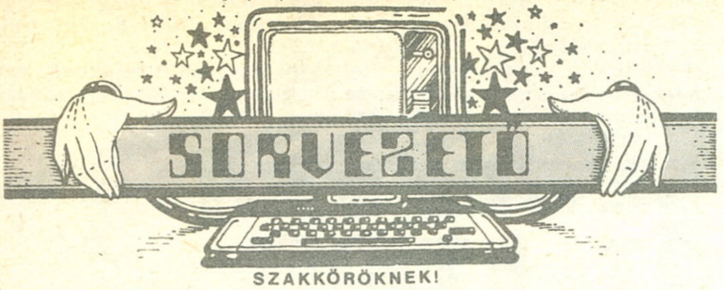
SZOFTVER ÖTLETEK



A következő gépi kódú programmal a ZX–81 képernyő pillanatnyi tartalmát lehet negatívba váltani. (Azaz a normál karakterek helyébe a grafikus megfelelőjük kerül és viszont.) Természetesen a későbbiekben kiírásra kerülő üzenetekre nincs hatással. A program a memóriában bárhol elhelyezhető. A POKE-oláshoz szükséges byte-okat (decimálisan) a táblázat jobb oldala tartalmazza:

	LD	HL, (400CH)	42	12	64
	LD	B,18H	6	24	
CIKL,	INC	HL	35		
	LD	A,(HL)	126		
	CP	76H	254	118	
	JR	Z,SORVEG	40	5	
	ADD	A,80H	198	128	
	LD	(HL),A	119		
	JR	CIKL	24	245	
SORVÉG,	DJNZ	CIKL	16	243	
	RET		201		

Bakos Sándor 6900 Makó, Kálvin u. 36.



SZAKKÖRÖKNEK!

System külső magnetofonhoz HT 1080Z

Valószínű – nem csak nekünk, hanem – sok más HT felhasználónak is régi vágya, hogy gépi kódú programokat tudjon beolvasni külső magnetofonról.

Mint ismeretes, az interpreter csak Basic programok betöltését teszi lehetővé (CLOAD = −2, "név"), gépi kódút nem. Így több kazettán lévő programunk vált a hosszas használat során (vagy a másik gépen készített felvétel miatt) betölthetetlenné. Környezetünkben van aki a külső magnetofon ilyen illesztését hardver úton próbálta megoldani – több-kevesebb sikerrel – de ezt az áramkörökhöz kevésbé értőknek javasolni nem merjük.

Ettől mi jóval egyszerűbb, és a gép átalakítását egyáltalán nem igénylő módszert ajánlunk:

Ha DISASSEMBLERREL visszafejtjük az interpretert, a 02B2H címtől kezdve (ez a SYSTEM parancs belépési pontja), megállapíthatjuk, hogy a *? után figyelmen kívül hagyja magnetofon-kijelölési szándékunkat, és automatikusan a belsőt indítja el. (A gépi kód visszafejtése igen bonyolult, ezzel részleteiben most nem is foglalkozunk.)

A beolvasás megoldható egy rövid programmal, amit a legpraktikusabban a monitorral írhatunk be olyan területre, amit a gép valószínűleg majd nem használ a rutin indításáig. Mivel a program maximum 20 – azaz húsz – byte (de lehet öttel kevesebb is), ez nagyon könnyen megtehető.

Az általunk választott memóriaterület a 6300H címtől kezdődik. (De lehet tetszőleges másik hely is.) Ide kell beírni a következő byte-okat:

21, 09, 63, 31, A0, 7F, C3, CE, 02, 3A,
23, CE, 32, 2C, 00, 00, 00, 00, 00, 00.

Az utolsó hat darab 00 helyére írhatjuk a beolvasandó program nevének megfelelő ASCII-kódokat. Nem fontos mind a hatot, hanem – amint a SYSTEM parancsnál megszoktuk elég annak az első néhány karakterét, és még egy darab 00-t.

Ha nem ismerjük a program nevét, vagy úgyis a kazettán soron levőt akarjuk beírni, akkor elég az első 00-t beírnunk. Miután ezzel elkészültünk, csatlakoztassunk külső magnetofont – a géppel kapott zsinór segítségével – a „TAPE RECORDER" bemenethez (a magnetofonba a fekete és kék vezetéket kell illeszteni). Ha előzőleg lenyomtuk volna, akkor engedjük fel az F1 gombot, és G6300-val (vagy ha másik címtől írtuk, akkor azzal) hívjuk meg a beírt rutint. A könnyebb megértés kedvéért a program mnemonikus formája:

```
LD HL, 6309H;    a kettőspont (23H) címe
LD SP, 7FA0H
JP 02CEH         ugrás a SYSTEM-re
```

A többi szám az alábbi BASIC-szavak kódja:

: – 2 ,

A listában ezután a program neve következik. Ez itt hat darab 00. (Ekkor tehát bármilyen névvel jelölt gépi kódú programot beolvashatunk.)

Aki nem túlságosan járatos a monitor használatában, annak közlünk egy BASIC-programot, amely elvégzi a megfelelő címek feltöltését, és egy kicsit „barátságosabb".

```
10 CLS:PRINT@12,"GEPI KODU PROGRAM BETOLTESE KULSO
   MAGNETOFONROL":
   Q$="":INPUT"A PROGRAM NEVE (0-6 KARAKTER) ";Q$:
   Q$=Q$+STRING$(6,0):Q$=LEFT$(Q$,6)
20 FOR I=25344 TO 25356:READ A:POKE I,A:NEXT:
   POKE 16526,0:POKE 16527,99
30 FOR I=1 TO 6:A=ASC(MID$(Q$,I,1)):POKE25356+I,A:NEXT
40 PRINT"ENGEDJE FEL AZ F1 GOMBOT ES
   INDITSA EL A KULSO MAGNETOFONT !":
   A=USR(0)
50 DATA 33,9,99,49,160,127,195,206,2,35,206,50,44
```

Ezt a néhány sort, természetesen nem kell állandóan beírni a gépbe. Célszerű felvenni egy kazettára és szükség esetén belső vagy külső magnetofonnal beolvashatjuk, ahányszor csak szükség van rá.

Tudomásunk szerint több olyan program van már forgalomban, amelynek beolvastatása eddig csak belső magnetofonról volt lehetséges (EDI, TBUG, LEVEL3, FORTH, PASCAL stb.). A Tudományszervezési és Informatikai Intézet is forgalmaz olyan oktatócsomagokat, amelyben gépi kódú programok vannak. Az említett eljárásunk jobb minőségű külső magnetofon használatát is lehetővé teszi, ha a belső magnóval a bevitel nem sikerülne.

A SYSTEM utáni pontos folyamat leírása helyett most elégedjen meg a tisztelt olvasó azzal, hogy működik a közölt lista, és azt csinálja, amit már rég szerettünk volna. Programunk felhasználásához jó munkát kívánunk!

Madarász József IV. o. tanuló
Kertész Béla szakfelügyelő
Tóth Árpád Gimnázium, Debrecen 4001 Pf. 49.

POSTA

BP 1986

Következő levélírónk inkognitóban szeretne maradni, ezért név nélkül közöljük levelét:

Tisztelt Szerkesztőség!

Egy kéréssel fordulok Önökhöz. Nyár óta egy Commodore 16-os számítógép boldog tulajdonosa vagyok. Eleinte problémákkal kellett szembenéznem, ugyanis a gép „nyelvezete" minden számomra ismert géptől eltér. Ezek a gondok szerencsére, hosszabb sikertelen próbálkozások után végül is megszűntek. Nagyon megszerettem a gépet és azóta szinte mindent, amit elképzeltem, meg is tudtam valósítani rajta. Két problémám azonban nem akar megoldódni. Az egyik az, hogy tudásomat, amit elsősorban szakkönyvekből és a géppel való rendszeres foglalkozás során szereztem, más gépeken nem vagy csak alig-alig tudom kamatoztatni. Kérdésem, várható-e, hogy ez a géptípus a 64-eshez hasonló népszerűségre tesz szert, vagy pedig ez a gép a Commodore cég egyik rosszul sikerült modellje? Másik kérdésem az, hogy hol tudnék gépi kódban írt játékprogramot szerezni?

1. Valószínűleg népszerű lesz a C–16 is. Ajánljuk figyelmébe a múlt havi BIT-LET-ünk vezércikket.
2. Játék- és egyéb programokhoz kétféleképpen lehet hozzájutni; külföldön pénzért vagy itthon más programokért cserébe, C–16 tulajdonosoktól. Ez utóbbival valószínűleg egyelőre olyan probléma van, hogy még nem túl sok tulajdonos és program van. Később mindkettő valószínűleg elég sok lesz.

A szerkesztő azért van,

hogy a lap olyan legyen,

amilyenek az olvasói!

PROGRAM BÖRZE

A szerkesztő 1985. májusának végén az alábbi felhívást intézte BIT-LET híveihez:
„...a hazai készítésű programok információáramlásában szívesen segítene lapunk is. Börzerovatunk ma még nincs, de éppen lehetne. Egy ilyen rovatban ki-ki közölhetné eladó programjainak nevét, a program kétsoros felhasználói leírását, a becsült eladási árat és azt a címet, ahol bővebb információt lehet kapni a programról. Hogy ilyen rovat lesz-e a BIT-LET-ben, ez elsősorban Önökön olvasókon, programozókon múlik."
Nos a véres kard körülhordozása máris eredménnyel járt. Így hát van szerencsénk bejelenteni a Börze rovat megszületését. Íme az első ajánlatok.

A BIT-LET-ben olvastam, hogy létre akarnak hozni egy olyan rovatot, amelyben programokat lehet eladni, cserebérélni. Ezt kiváló ötletnek tartom, de vegyék bele ebbe a rovatba az amatőrök által készített programok árusítását is. Ezzel – szerintem – nagyobb sikert aratna a szóban forgó rovatuk. De – hogy mindez ne legyen üres szóbeszéd –, szeretnék én is beküldeni egy ilyen programajánlatot.
Ajánlott program: ***LEVEL 4 BASIC – bővítő.*** Gépi kódú HT–1080Z iskolaszámítógépre. Rendkívüli gyorsaság! Aktív képernyőhasználat. Érték: megegyezés szerint!
További felvilágosítás és részletes leírás: Tamás Ferenc, Nagykanizsa, Bartók B. u. 5. 2/2/3 8800

A C64-hez használt legegyszerűbb printereken (MPS801, SEIKOSHA GP–100VC, 1525) megoldottuk a karakterek tetszés szerinti változtatását tisztán szoftver úton. A KERNAL-rutinok manipulálásával gépi kódú szubrutinokat illesztettünk az operációs rendszerhez, így a gép normál módon használható és programozható, csak éppen bizonyos karakterek helyett mást ír a képernyőre és a papírra. És ami új a hasonló programokkal szemben: hogy melyik billentyű milyen jelet jelentsen, azt a felhasználó maga szerkesztheti meg egy EDITOR program segítségével, mely a screen-editorhoz hasonló kurzoros módszerrel és sokféle szolgáltatással teszi játékosan könynyűvé a megfelelő dot-mátrixok kialakítását. Így akár többféle karakterkészlet is tárolható, s munka előtt a szükséges behívható egy LOADER program segítségével, mely azt is megkérdezi, hol foglaljon le helyet a memóriában a képernyő és a karakterkészlet számára. A C000-val kezdődő területet választva az is elérhető, hogy a szabad BASIC-terület nem csökken, hanem még 1 kByte-tal nő is.
Aki tehát olyan lehetőséget keres, hogy ékezetes betűket vagy más jeleket is tudjon nyomtatni a printere, az forduljon hozzánk, mert jobb programot kap olcsóbban, mint más hasonló termékek.
A program neve legyen mondjuk ***CLEVER 801, az ára 10 000 Ft*** mágneslemezzel és precíz leírással együtt. Ha megrendelőlevelet küldenek a címünkre, mi személyesen visszük el a programot, s a használatbavételhez útmutatót is adunk. Címünk, ahonnan felvilágosítást vagy bemutatót is lehet kérni:
REGISZTER GM
2120 Dunakeszi, Garas u. 10. IV. 14.
Tel.: 27/41-636

A Commodore Szekció elkezdi a „klublemezek" akciót. A „klublemezek" a VC 1541 lemezegységekhez készített kétoldalas lemezeken kerülnek forgalomba. A lemez egyik oldala a felhasználó részére szabadon programozható üres, de formált, míg a másik oldalon mintegy 100 blokk terjedelemben trükkök, a géphasználatot segítő professzionális alkalmazási és játékprogramok találhatók. A lemezek ára, a célnak megfelelően, lényegesen alacsonyabb, mint az itthon megszokott, és a lemezeken csak Magyarországon mások által szerzői jogi védelemben nem részesített programok lesznek. Különleges szolgáltatás az is, hogy a használathoz szükséges hardvereszközöket, (botkormány) is hozzá adjuk.
Az eddig meglévő programok a következők:
- ***Tesztkérdések*** (36 blokk): Oktatási segédlet, mellyel tesztkérdések (állományonként maximum 100) állíthatók össze (lemezenként maximum 15 lehet). Ezeket lehet: szerkeszteni, lemezre venni, a tanulók válaszai alapján adott osztályzatokat is lemezre lehet vinni; a kérdéseket lehet: cserélni, javítani, új kérdésekkel kiegészíteni, lemezről betölteni, a lemezek lemezazonosítókkal is elláthatók és törölhetők.
- ***Szótárkészítő*** (15 blokk): szavak, évszámok, kifejezések stb. begyakorlására szolgáló program. A szótárt lehet: bővíteni, javítani, lemezre vagy szalagra vinni, onnan visszahozni, a tanulást ellenőrizni. A rendszer a rossz válaszokat gyűjti, és végén kiírja.
- ***HI-RES képnyomtató*** (1 blokk): A nagyfelbontású képet képernyőre, vagy VC 1525, vagy MPS 801 típusú nyomtatóra viszi.
- ***Taxi*** (46 blokk): Közlekedési játék botkormánnyal. A játékos „taxijával" a diszpécser illetve utas által megadott helyekről más helyekre visz utasokat. Az útvonalat a képernyőn lévő térkép alaján maga választja meg. Útközben forgalmi lámpák, keresztező járművek akadályozzák, illetve a megtehető út hosszát az üzemanyagfogyás korlátozza.
- ***Függvényközelítés*** (21 blokk): Az adatokat billentyűzetről vagy lemezről fogadó, az eredményeket a képernyőre, vagy nyomtatóra vivő mérésadat-feldolgozó program. Tetszés szerinti számú változójú automatikus függvényközelítést végez. A szokásos megoldásoktól eltérően, nemcsak a függvény állandóit optimalizálja, hanem magát a függvényformát is.
- ***Grafikus nyelv*** (47 blokk): Egyszerű kis térfoglalású BASIC-ből is hívható grafikus nyelv, pont-vonal rajzoló és törlő, valamint képernyőtörlő utasítással. Nagyfelbontású grafikát használ.
- ***Szűkített LOGO*** (45 blokk): A LOGO-nyelv „teknőc-grafikának" 30 parancsát valósítja meg (beleértve az eljárások használatát). Önmagát az eljárást sem közvetlenül, sem közvetve nem hívhatja meg. Képszerkesztést, névlistázást, nyomtatást, névváltoztatást, eljárások lemezre vagy kazettára vitelét és betöltését, lemezen törlést végez.
- ***Assembler – disassembler*** (16 blokk): Egymenetes fordítást (másodlagos utasítások, pszeudo-operátorok) használatát lehetővé tevő, egyúttal a használót az assembler nyelvre tanító hexa- és decimális bemenetű assembler.
- ***Gépi kódú programkészítő segédlet*** (12 blokk): Programbeírást, soron belüli javítást, programmentést, betöltést, összekapcsolást lehetővé tevő eszköz.
- ***Lemezprogramot automatikusan indító rutin*** (11 blokk): A LOAD. . .8,1 forma alkalmazását teszi lehetővé a felhasználó által írt programnál.
- ***f – billentyűk programozására*** szolgáló program (8 blokk): lehetővé teszi ezen a billentyűk programozását, és a program mentését.
- ***Ugorj*** (14 blokk): Űrinváziós típusú játék.
- ***Két szólam*** (28 blokk): A billentyűzet két felét, mint független szólamot vivő billentyűzetet használ.
- ***MERGE*** (5 blokk): Lemezen levő programot kapcsol a tárban levőhöz.
- ***RENUMBER*** (2 blokk): Megadható növekményű és kezdőértékű újraszámozó (SIMON'S BASIC-tól eltérően az ugróutasítások címeit is megfelelően módosítja).
- ***Képvizsgáló*** (12 blokk): KOALA PAINT és egyéb képvizsgálat, áthelyezés, beolvasás, átfordítás.
- ***BASIC és gépi program összekapcsoló*** (16 blokk): Tömbkiválasztó és kapcsoló.
- ***Függvényforma betöltő BASIC-programba*** (5 blokk): Tetszés szerinti, de nem túl bonyolult függvényt lehet bevinni az INPUT-utasítással.
- ***GET-használó*** (4 blokk): A gép a GET-használatával beállítható hosszúságú maximum 191 szám, vagy alfanumerikus karaktersorozatot fogad el.
- ***SORT-utasítás*** (18 blokk): BASIC-ből hívható gépi kódú sorbarendező. A rendezés növekvő vagy csökkenő rendben és vagy az adaton belül kijelölt paraméter szerint történik.
- ***Állománykezelő*** (30 blokk): Nagy (maximum 30 kbyte) állományok kezelésére szolgáló program, az adatmező maximum 80 karakter. Alkalmas: új rekord, állomány készítésére, összekapcsolására, törlésére, üres állomány készítésére, vizsgálatára, az adatmező rendezésére, alfabetikusan növekvő rendbe, módosításra.
- ***MULTIPROCESSING*** (14 blokk): Használatával több program futhat egyidejűleg.

Egy lemez ára botkormánnyal együtt 5000 Ft.
Ügyintéző: Ollé Tibor 14. sz. ÜMK. Budapest VII., Rákóczi út 8/a.
Ez az ár minden lemezre érvényes a Mikromagazin 85/2 számának 28. oldalán ismertetett program kivételével. Ez a professzionális fejlesztő program, amelyhez hasonlót nem ismerünk, ára: 20 000 Ft.

A PRIMO-nyerő 3. feladat megoldása:

a) Ilyen derékszögű háromszög nincs. Legyen ugyanis a háromszög két oldala a és b úgy, hogy $a^2-b^2 = 15$ teljesüljön. Ekkor $(a+b) * (a-b) = 15$, s mivel a és b pozitív egész, ezért vagy

$a+b = 5$, vagy $a+b = 15$

$a-b = 3$, $a-b = 1$

Az első esetben a = 4, b = 1, s mivel se 16+1 = 17, se 16−1 = 15 nem négyzetszám, ezért ezek nem lehetnek egy egész oldalú derékszögű háromszög oldalai.

A második esetben a = 8, b = 7, és se 64+49 = 113, se 64−49 = 15 nem négyzetszámok, így megint arra jutunk, hogy ezek sem lehetnek egy megfelelő háromszög oldalai. Mivel a 15 máshogy nem bontható fel egy pozitív egész és egy egész szám szorzatára, bizonyításunkat befejeztük.

b) $1+2+\ldots+1526 = \frac{1526 \cdot 1527}{2}$ minek a prímtényezős felbontása 3.7.109.509. Feltehetjük, hogy a kapitány 100 évesnél fiatalabb, s 18 évesnél idősebb, így csakis 21 éves lehet. (Lehetséges, hogy ilyen fiatal kapitányok nem nyüzsögnek a világon, de a mi kapitányunk papája magas rangú személyiség, s ő maga nagyon tehetséges volt, külhonban és évösszevonásokkal végezte iskoláit. Mindenesetre megfelelő indoklással a 109 éves kapitányt is elfogadjuk helyes megfejtésnek).

2 Gépnyerő

HT 2080Z
PRIMO

Ígéretünkhöz híven a Tudományszervezési és Informatikai Intézet és a Bit-let újabb közös – háromfordulós szakköri pályázatot indít, két kategóriában. A két kategóriát azért tartjuk szükségesnek, mert előző pályázatunk során kiderült, hogy egy sor olyan szintű szakkör működik már az országban, amelynek színvonala lényegesen magasabb a többi pályázatunkban számba jöhető közösségnél. A kategorizálást tehát kizárólag az eddigi két pályázatunkon elért eredmények alapján végeztük. Ily módon ***első kategóriában indulhatnak a következő iskolák szakkörei: ELTE Apáczai Gimn., Bp. • Berze Nagy J. Gimn., Gyöngyös • Berzsenyi D. Gimn., Bp. • 1. sz. Ipari Szakközépisk., Miskolc • Fazekas M. Gimn., Debrecen • Földes F. Gimn., Miskolc • 600. sz. Ipari Szakmunkásképző Int., Szeged • JATE Ságvári Gimn., Szeged • Katona J. Gimn., Kecskemét • Kolos R. Finommech. Szki., Bp. • Leövey K. Gimn., Pécs • Móricz Zs. Gimn., Szentendre • Móricz Zs. Gimn., Tiszakécske • Piarista Gimn., Bp. • Roth Gy. Erdőgazd. Szki., Sopron • Svetits Gimn., Debrecen • Szalvay M. Úttörőház, Kecskemét • Széchenyi I. Gimn., Sopron • Varga K. Gimn., Szolnok • Vörösmarty M. Gimn., Érd***

Második kategóriában indulhatnak: minden olyan általános iskola, középfokú intézmény, úttörőház és HT-klub szakkörei, csapatai, mely intézmények nincsenek benne az előbbi felsorolásban.

Pályázati feltételek:

1. Minden programot HT–1080Z (HT–2080Z) gépre kell elkészíteni, s kazettán kell beküldeni, legalább háromszor felvéve, lehetőleg különböző gépeken. A végleges programok kazettára mentése után mindhárom példányt olvastassuk be a gépbe (ha lehet másikba, mint amelyiken felvettük), s újra teszteljük le! Ha egy program egyik változatát se tudjuk betölteni 1–2–3-szori próbálkozás után sem, illetve ha a program hibás, nem tudjuk értékelni, s így 0 pontot kap.

2. A kazettára, vagy a dobozában valamilyen papírra feltétlenül írjuk fel, hogy a kazettán mettől meddig, milyen programok találhatók, s hogy a felvétel régi vagy új gépen készült-e?

3. Ha a kazetta betöltése előtt a gépen valamilyen tennivaló van (bővítés kikapcsolása, RAMTOP állítása), ezt jelezzék ezen a helyen is.

4. A pályázatnak ezen kívül tartalmaznia kell egy leírást, melyen fel van tüntetve az iskola pontos címe és a szakkör neve (s ez minden forulóban ugyanaz legyen!), s melyben röviden leírják a program használatát és működését (algoritmusát) szöveggel.

5. A pályázatokat a beküldési határidő napján 24 óráig kell postára adni a következő címre:
TUDOMÁNYSZERVEZÉSI ÉS INFORMATIKAI INTÉZET, MIHÁLYFI JÁNOS 1111 BUDAPEST, EGRI JÓZSEF U. 1–9. E ÉPÜLET

Az olyan pályázatokat, melyek a fenti feltételeknek nem tesznek maradéktalanul eleget, nem értékeljük.

A pályázat két kisebb és egy nagyobb feladatból fog állni, a kisebbek 50, a nagyobb feladat 100 pontot fog érni. Ebben a számban az első kisebb feladatot közöljük, a következőben a nagy feladatot, s a harmadik fordulóban a másik kicsit; a második és a harmadik feladat beküldési határideje azonos lesz.

A pályázat díjai: az I. kategóriában az első díj ismét egy 64 K-s HT-gép, a második kategóriában pedig egy PRIMO!

S most lássuk az első feladatot:

készítsünk programot, mely jól játssza a közismert MALOM nevű játékot (leírását I. pl. Zdzislaw Nowok: 50 táblás játék c. könyvében a 12. oldalon). Az egyik játékos a gép legyen, s „ő" adminisztrálja a játékot, jelenítse meg a pillanatnyi állást, és lehetőleg nyerjen. Ezenkívül azt is megkívánjuk, hogy a program képes legyen bizonyos fokú tanulásra, azaz a játékok során egyre okosabb legyen, egy csapdába ne essen bele kétszer. Fontos elbírálási szempont lesz, hogy milyen jól tud tanulni a gép, a játszmák közbeni információkat milyen jól hasznosítja, s hogy milyen gyors a program. (Az ember nem szívesen ül le olyan programmal játszani, mely minden lépésen egy negyedórát gondolkodik). A leírásnak tartalmaznia kell a program játszási stratégiáját és tanulási mechanizmusát! Sikeres vetélkedést!



Kérjük levágni és a levélre felragasztani! Beküldési határidő október 30

Szerkesztőségünk címe: ÖTLET Budapest 1986. Szerkesztő: Angyalosi László. Művészeti szerkesztő: Pribelszky Pál. Tördelő: Domokos Imre. Hírrovat: Csányi György. Vállató: Pogány György. Szervező: Székely Jenő. Posta: Halász Péter